TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN MÔN PHÂN TÍCH THIẾT KẾ VÀ GIẢI THUẬT

ĐỀ XUẤT THUẬT TOÁN

Giảng viên dạy lý thuyết: **TS. Nguyễn Chí Thiện**

Giảng viên dạy thực hành: **ThS. Võ Đức Vĩnh**

Người thực hiện: **Hồ Phương Hiếu - 51303286**

Lớp : **13050303**

Khóa : **17**

THÀNH PHỐ HỒ CHÍ MINH, 2016

LỜI CẢM ƠN

Em xin chân thành cảm ơn, thầy Nguyễn Chí Thiện đã hướng dẫn em phần lý thuyết môn Phân tích thiết kế và giải thuật, cảm ơn thầy đã ra đề tài "Đề xuất thuật toán" để em có thể tìm hiểu và hiểu sâu hơn các vấn đề về các loại giải thuật. Đồng thời em xin chân thành cảm ơn thầy Võ Đức Vĩnh đã hướng dẫn em trong buổi thực hành để em có thể cài đặt các giải thuật trong thực tế.

Tuy đã cố gắng hoàn thành bài toán nhưng có thể không tránh khỏi sai sót và hạn chế trong quá trình lam bài. Kính mong quý thầy cô bỏ qua và điều chỉnh lại để bài lam đạt kết quả tốt hơn và em có thể khắc phục được vấn đề ở những môn học tiếp theo.

BÀI TẬP LỚN ĐƯỢC HOÀN THÀNH
TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi và được sự hướng dẫn của TS Nguyễn Chí Thiện; Võ Đức Vĩnh. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung bài tập lớn của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 01 tháng 04 năm 2016

Tác giả

(ký tên và ghi rõ họ tên)

Hồ Phương Hiếu

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm 2016

(ký và ghi họ tên)

Phần đánh giá của GV chấm bài

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm 2016

(ký và ghi họ tên)

TÓM TẮT

Đồ án gồm có hai phần :

1. Phân tích và đề xuất một thuật toán để tối đa hóa về yêu cầu khách hàng và lợi nhuận người bán đạt được.

   Trong phần này ta sẽ trình bày thuật toán từ giải thuật tham lam cho việc tối đa về số lượng khách hàng yêu cầu và phương pháp quy hoạch động để tối đa lợi nhuận cho người bán.
2. Tối đa hóa lợi nhuận từ một thuật toán tất định và thuật toán ngẫu nhiên cho việc bán nhà.

MỤC LỤC

MỤC LỤC 6

I. Phân tích và đề xuất một thuật toán tối đa hóa(cho bài toán bán đất):..... 6

1. Tối đa hóa số lượng yêu cầu bằng giải thuật tham lam: 7

2. Tối đa hóa số tiền bán được bằng phương pháp quy hoạch động: 8

II. Đề xuất thuật toán để tối đa hóa lợi nhuận(cho bài toán bán nhà):..... 13

1. Trường hợp bán cho ngân hàng: 13

2. Trường hợp bán cho người mua: 14

TÀI LIỆU THAM KHẢO..... 15

I. Phân tích và đề xuất một thuật toán tối đa hóa(cho bài toán bán đất):

1. Tối đa hóa số lượng yêu cầu bằng giải thuật tham lam:

Để tối đa số lượng hóa số lượng yêu cầu của khách hàng có thể đáp ứng thì chúng ta có thể sử dụng giải thuật tham lam với chọn lựa là những yêu cầu nào có tọa độ thứ 2 nhỏ hơn thì sẽ được ưu tiên chọn trước:

a. Thiết kế:

Giải thuật (Array T:1…n)

Begin

1. Array Result:1..n
2. $T \rightarrow$ SortByT2
3. Result add T[0]
4. For $i \rightarrow 1\ to\ n$ do

   Begin For
5. IF T[i] is compatible with Result then

   Begin IF
6. Result add T[i]

   End IF

   End For
7. Return Result

   End

b. Cài đặt:

```
public static boolean OrderByPointB(List<ARC> lstARC) {
      if (lstARC == null || lstARC.size() == 0)
          return false;
      ;
```

```
        Collections.sort(lstARC);
        return true;
    }
    public static List<ARC> BestNumberOfRequest(List<ARC>
lstARC) {
        OrderByPointB(lstARC);
        List<ARC> lstResult = new LinkedList<ARC>();
        lstResult.add(lstARC.get(0));
        for (int i = 1; i < lstARC.size(); i++) {
            ARC lastResult =
lstResult.get(lstResult.size() - 1);
            ARC current = lstARC.get(i);
            if (current.pointA >= lastResult.pointB) {
                lstResult.add(current);
            }
        }
        return lstResult;
    }
```

2. Tối đa hóa số tiền bán được bằng phương pháp quy hoạch động:

Để tối đa hóa số tiền kiếm được thì chúng ta có thể sử dụng phương pháp quy hoạch động để giải quyết bài toán này.

Công thức QHĐ:

Hàm mục tiêu : f = độ dài dãy con.

Vì độ dài dãy con chỉ phụ thuộc vào 1 yếu tố là dãy ban đầu nên bảng phương án là bảng một chiều. Gọi L(i) là độ dài dãy con tăng dài nhất, các phần tử

lấy trong miền từ a1 đến ai và phần tử cuối cùng là ai. Nhận xét với cách làm này ta đã chia 1 bài toán lớn (dãy con của n số) thành các bài toán con cùng kiểu có kích thước nhỏ hơn (dãy con của dãy i số). Vấn đề là công thức truy hồi để phối hợp kết quả của các bài toán con. Ta có công thức QHĐ để tính L(i) như sau:

- L(1) = 1. (Hiển nhiên)
- L(i) = max(1, L(j)+1 với mọi phần tử j: 0<j<i và aj≤ai).

Tính L(i) : phần tử đang được xét là ai .Ta tìm đến phần tử aj < ai có L(j) lớn nhất. Khi đó nếu bổ sung ai vào sau dãy con... aj ta sẽ được dãy con tăng dần dài nhất xét từ a1...ai.

a. Thiết kế:

Bảng phương án là một mảng một chiều L để lưu trữ các giá trị của hàm QHĐ L(i). Đoạn chương trình tính các giá trị của mảng L như sau:

```
For i := 1 to n do
Begin
L[i] := 1
for j:=1 to i-1 do
if (a[j]<=a[i]) and (L[i]<L[j]+1) then
L[i]:=L[j]+1;
End
```

Như vậy chi phí không gian của bài toán là O(n), chi phí thời gian là O(n2). Có một phương pháp cài đặt tốt hơn so với phương pháp trên, cho chi phí thời gian là O(nlogn)

Vậy để áp dụng công thức quy hoạch động vào để giải bài toán vừa nêu trên thì ta phải chia bài toán lớn ra làm nhiều bài toán nhỏ như sau:

Ta sẽ cho L[i] bằng mức giá của yêu cầu thứ i, sau đó dựa vào đó ta có thể tính toán được một cách nhanh nhất những yêu cầu phía sau bằng cách áp dụng kết quả của những yêu cầu phía trước.

Bài toán này giống như câu a nhưng chỉ khác vấn đề là có thêm mức giá để giải quyết, vậy thì chúng ta vẫn sắp xếp ưu tiên theo θ_{i2} nhằm mục đích chọn được những khoảng đất hợp lý nhất.

Từ đó ta sẽ tính lần lượt yêu cầu của từng người

Yêu cầu ưu tiên sau thì xem xét những yêu cầu được ưu tiên trước đó để xem khi kết hợp vào nó có thỏa được điều kiện là những mãnh đất không chồng chất lên nhau hay không

Giá trị L[i] sẽ cho ta biết được đối với yêu cầu thứ i thì mức giá tối đa mà từ yêu cầu thứ i trở về trước là bao nhiêu. Dựa vào đó ta có thể chọn được yêu cầu mà có nó kết hợp với những yêu cầu trước có được mức giá tổng cao nhất.

b. Mã giả:

```
Algorithms Tối_đa_lợi_nhuận(ArrayT: 1..n)
    Begin
        1. Array Result
        2. T ≔ OrderBy(T → θ2)
        3. For i ← 0 to n
        4. L[i] ≔ T[i] → value
        5. For j ← 0 to i − 1
        6. If T[j] → θ2 ≤ T[i] → θ1 and L[i] < L[j] + T[i] → value
        7. L[i] = L[j] + T[i] → value
```

8. Result ≔ tracking L with Best Value
9. Return Result

End

c. Cài đặt:

```
public static List<ARC> BestGrossProfit(List<ARC> lstARC)
{
	List<ARC> lstResult = new LinkedList<ARC>();
	int n = lstARC.size();
	int[] V = new int[n];
	int[] L = new int[100];
	V[0] = 0;
	for (int i = 0; i < n; i++) {
		L[i] = lstARC.get(i).value;
		for (int j = 0; j < i; j++) {
			if ((lstARC.get(j).pointB <=
lstARC.get(i).pointA)
				&& (L[i] < L[j] +
lstARC.get(i).value)) {
				L[i] = L[j] + lstARC.get(i).value;
				V[i] = j;
			}
		}
	}
	int index = max(L);
	lstResult.add(lstARC.get(index));
```

```
        for (int i = index - 1; i >= 0; i--) {
            if ((L[index] - lstARC.get(index).value ==
L[i])) {
                lstResult.add(lstARC.get(i));
                index = i;
            }
        }
        return lstResult;
    }
    static int max(int[] L) {
        int max = L[0];
        int index = 0;
        for (int i = 1; i < L.length; i++) {
            if (L[i] > max) {
                max = L[i];
                index = i;
            }
        }
        return index;
    }
}
```

II. Đề xuất thuật toán để tối đa hóa lợi nhuận(cho bài toán bán nhà):

Theo đề yêu cầu thì đối với bất kỳ danh sách đề nghị nào, trong đó x là lời đề nghị tốt nhất, thuật toán luôn chọn được lời đề nghị có giá trị không ít hơn x/$\sqrt{M}$ (trăm triệu đồng).

Đề bài đưa ra bài toán với vấn đề giải quyết ở đây là làm sao cho giá bán không ít hơn x/$\sqrt{M}$, để giải quyết được vấn đề đó thì chúng ta cần xem.

Tại mỗi thời điểm người bán chỉ nhận được 1 yêu cầu từ người mua để ra quyết định bán hay không. Người bán mặc dù có giá bán mong muốn là , nhưng có một giá ngưỡng nào
đó (lẽ dĩ nhiên ngưỡng đó ≤ $\sqrt{M}$), nếu người mua ra giá đạt được ngưỡng đó thì người bán sẽ bán. Nếu chờ đợi mãi, hết thời gian rồi mà không ai trả được giá vượt ngưỡng thì phải chấp nhận bán cho ngân hàng với giá là 1.

Xét vấn đề: sẽ xảy ra hai trường hợp

1. Trường hợp bán cho ngân hàng:

Để bán cho ngân hàng thì tức là ta không tìm ra được giá bán phù hợp, vậy giả sử sau khi ta xem xét hết những lời chào mua nhà những vẫn chưa tìm ra giá phù hợp thì ta bán cho ngân hàng với giá 1 trăm triệu đồng và 1 trăm triệu đồng thỏa yêu cầu x/$\sqrt{M}$

$$\text{Tức là } 1 \geq \frac{x}{\sqrt{M}} \Leftrightarrow \frac{\sqrt{M}}{\sqrt{M}} \geq \frac{x}{\sqrt{M}} \Leftrightarrow \sqrt{M} \geq x \ (I)$$

Điều đó chứng tỏ rằng trong trường hợp giá bán tốt nhất nhỏ hơn $\sqrt{M}$ thì chúng ta có quyền bán cho ngân hàng mặc dù là có thể chọn được giá bán > 1 trăm triệu đồng.

2. Trường hợp bán cho người mua:

Từ trường hợp 1 ta có thể suy luận ra được rằng nếu bán được cho người mua thì giá bán (g) $g > \sqrt{M} = \frac{M}{\sqrt{M}} \geq \frac{x}{\sqrt{M}}$

Điều đó có nghĩa là $\forall g, g > \sqrt{M} \Longrightarrow g \geq \frac{x}{\sqrt{M}}$ (II)

Vậy từ (I)& (II) ta có thể kết luận rằng nếu $g > \sqrt{M}$ thì ta sẽ bán cho người mua còn nếu không tìm được thì ta sẽ bán cho ngân hàng

```
Mã giả
Algorithms Bán_đất(Array:1…n)
Begin
For i = 1 to n do
Begin
If i → value ≥ √M then
Return i
End For
Return  1;
End
```

TÀI LIỆU THAM KHẢO

Slide: Bài giảng phân tích thiết kế và giải thuật, thầy Nguyễn Chí Thiện, giảng viên khoa CNTT, đại học Tôn Đức Thắng.

Slide :Bài giảng của TS. Nguyễn Minh Tuấn, giảng viên khoa CNTT, đại học Tôn Đức Thắng.

Website: https://en.wikipedia.org/wiki/Sorting_algorithm

https://sites.google.com/site/nguoisaigonblog/giai-thuat-sap-xep-tinh-te

https://voer.edu.vn/m/thuat-toan-sap-xep/ff96502c

https://voer.edu.vn/c/phan-tich-thoi-gian-thuc-hien-thuat-toan/60bbf7d3/a7204439

Ebook https://uet.vnu.edu.vn/tltk/Learning/File_PDF/Cautrucdulieu.pdf